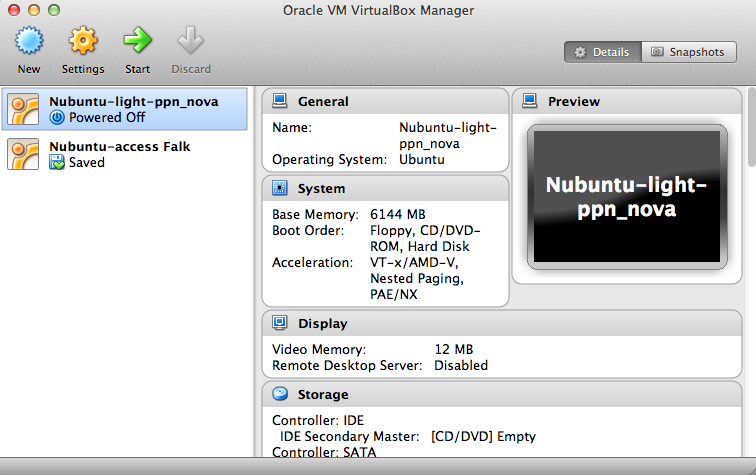
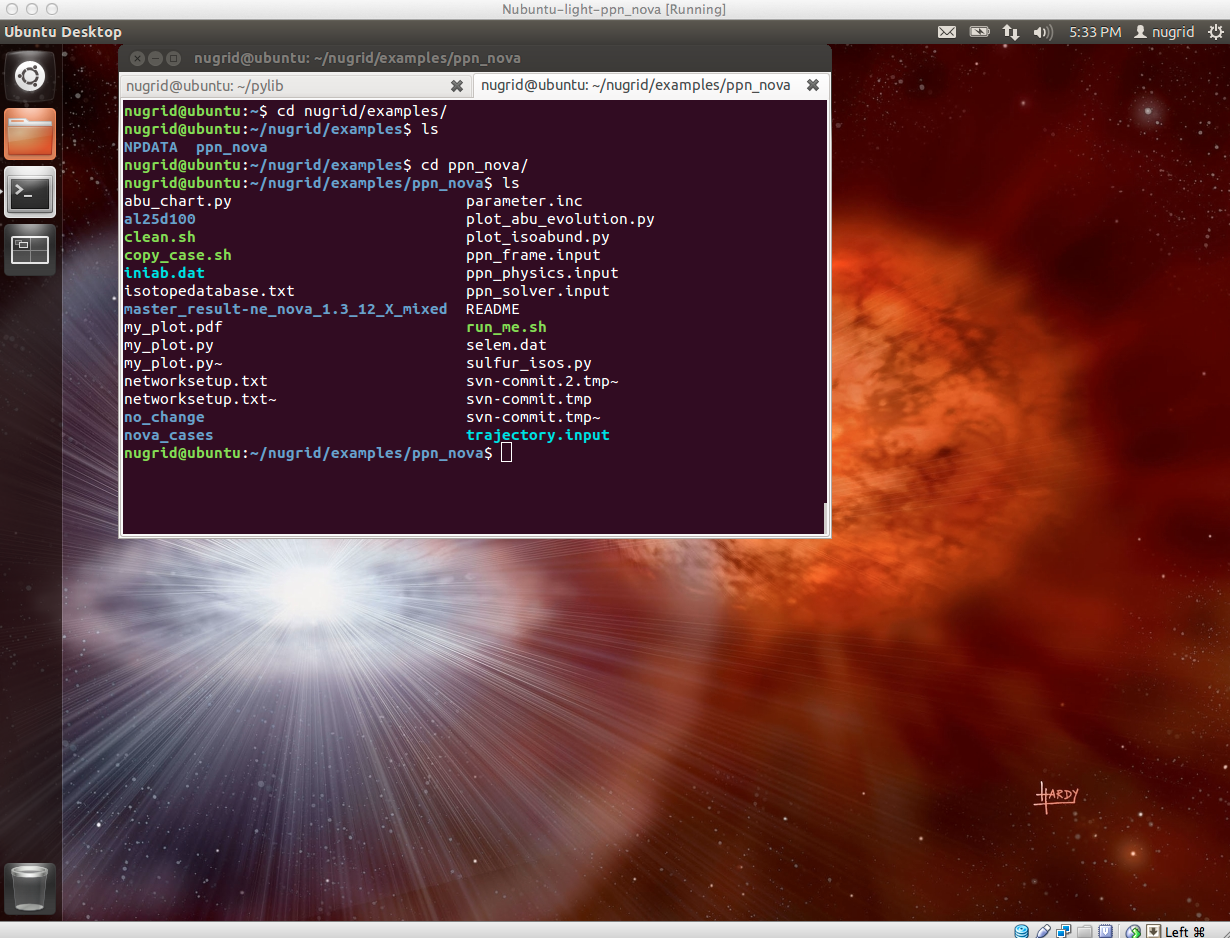
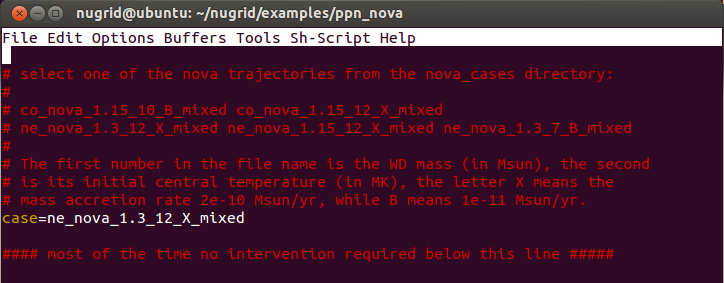
# A step-by-step guide to perform a reaction rate sensitivity test with the NuGrid-JINA-TRIUMF nova ppn framework

The goal of this HowTo is to demonstrate how to perform a simple task using the NuGrid nova ppn framework in a VirtualBox machine. Specifically we want ton show how Fig.15 in Denissenkov etal. 2013 (arXiv:1303.6265) can be made. It is assumed that the VirtualBox software (<https://www.virtualbox.org>) is installed and the Nubuntu-light-ppn\_nova machine has been downloaded and started. In order to open a virtual box machine for the first time double click on the vbox file. The VirtualBox Manager window would then look like this:

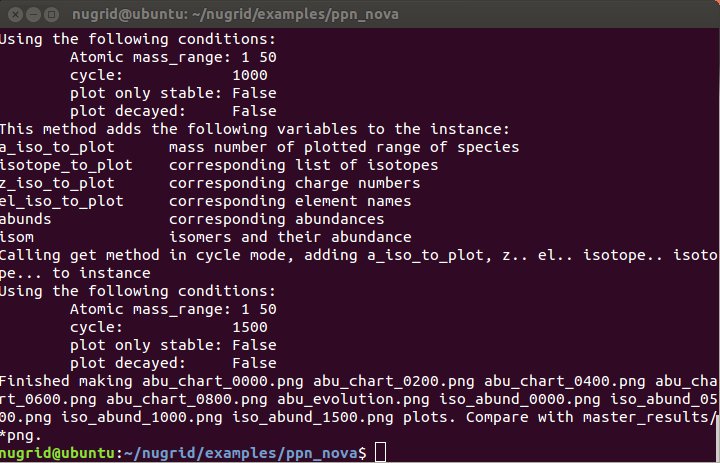
After starting the Nubuntu-light-ppn\_nova machine, and logging in (user name and passwd are ‘nugrid’) we can open a terminal in the virtual machine and navigate to the location of the ppn nova run directory on the command line:



In that directory you find a README file with useful information. In order to perform a standard run open the run script run\_me.sh and select one of the available nova trajectories (this may be updated in the version you are using):



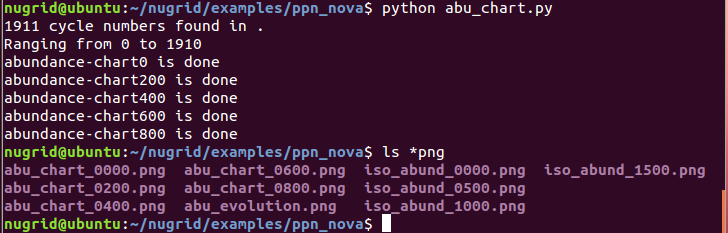
Then just execute the run\_me.sh script. After a couple of minutes you will get some output indicating that the standard plots are automatically performed:



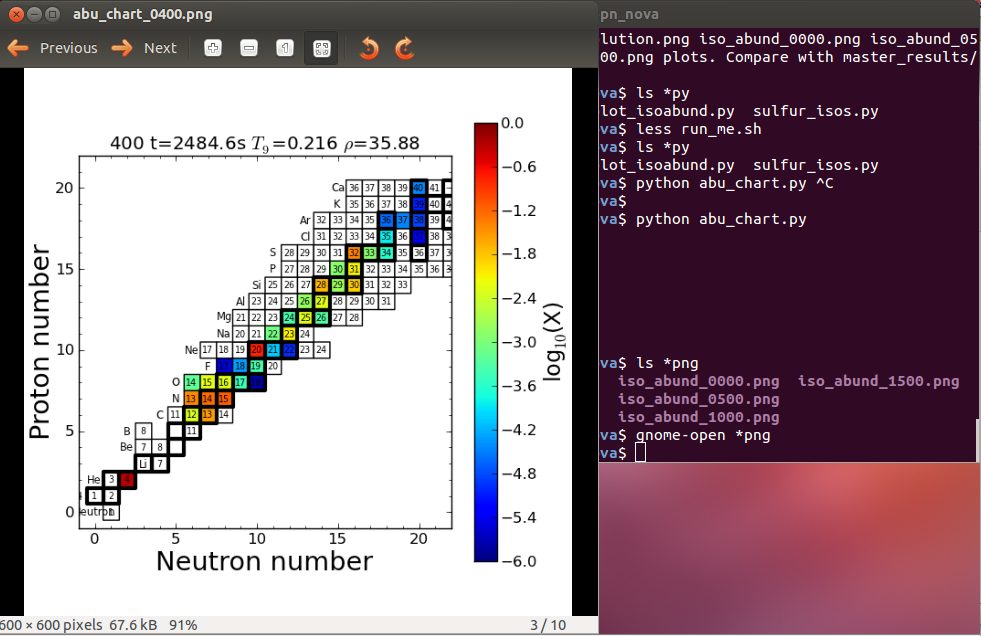
These standard plots are created using the \*.py files, which can be executed in batch mode as an argument to python:



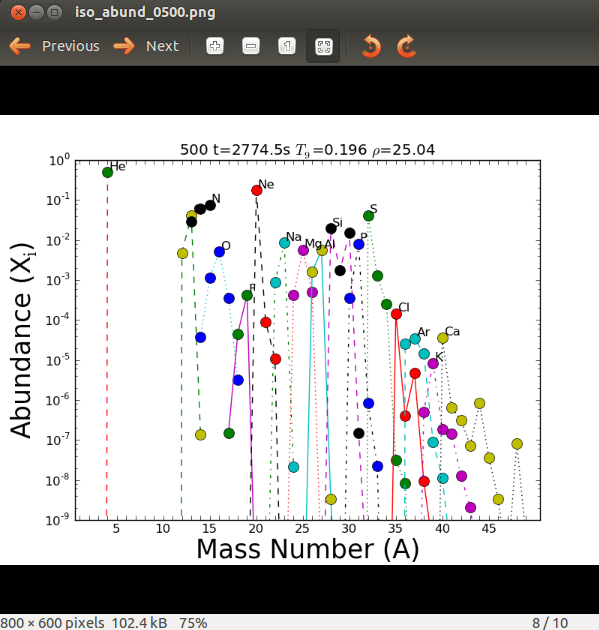
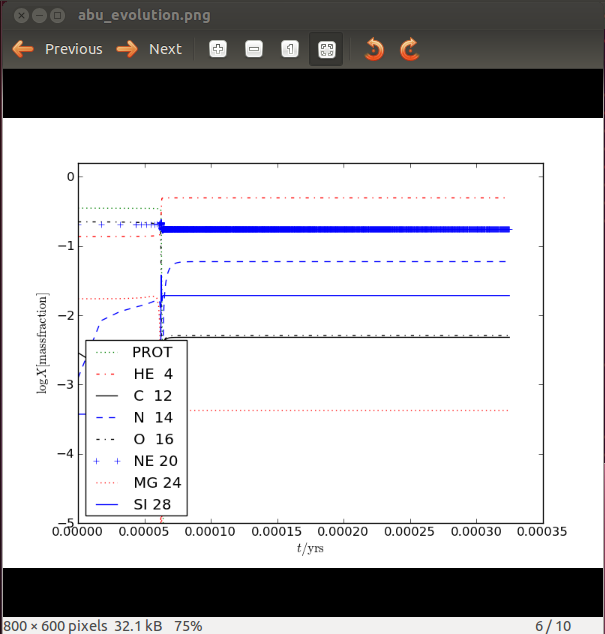
This and the other \*py scripts produce the \*png image files:



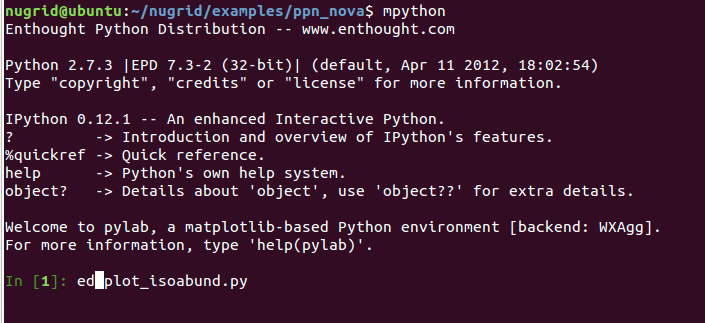
The plots can be viewed with the gnome-open command:

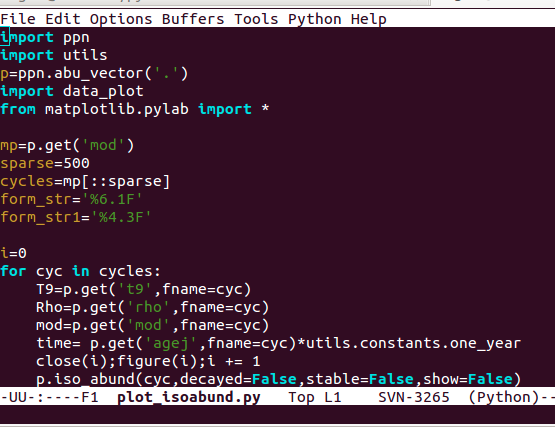


In addition to the abundance chart plots for a number of cycles there are abundance distribution plots as well as abundance evolution plots.



An easy way to work with these plots interactively is to specify the EDITOR environment variable (defaults to ‘emacs –nw’ which is emacs in terminal mode) and then use the ‘ed’ method interactively in ipython –pylab (which is aliased as mpython):

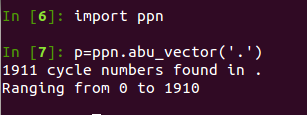


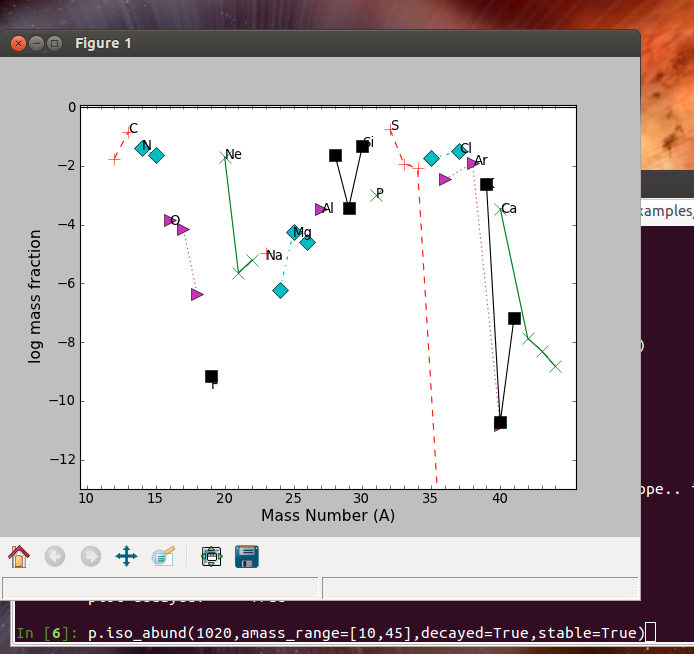
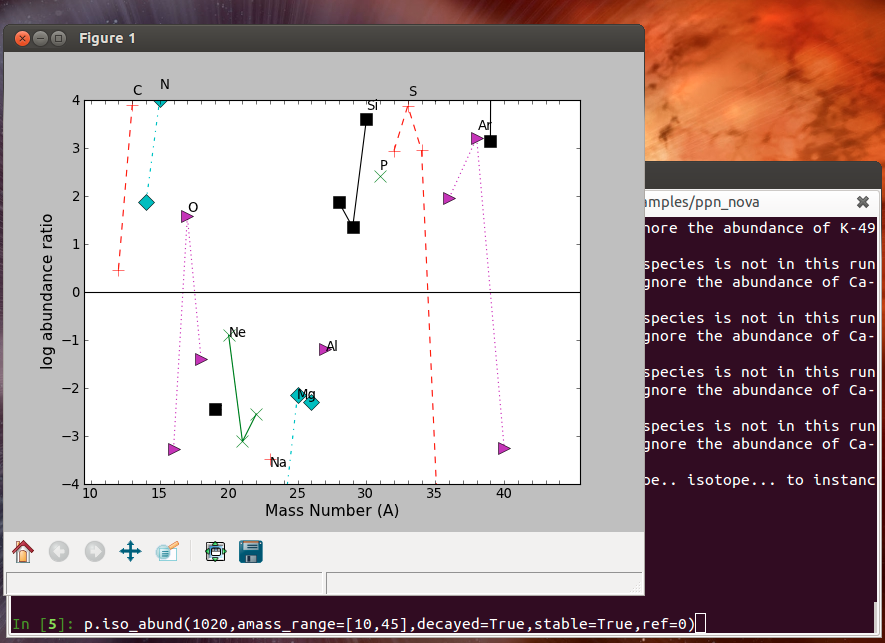


Once saved and exited the script will be executed as if typed in the command line, and all names and variables are available in the current session.

Let’s go through the individual steps performed in the script one by one (start an ipython session with ipython –-pylab or the corresponding alias mpython, the ppn module source code can be viewed in $HOME/pylib/ppn.py):

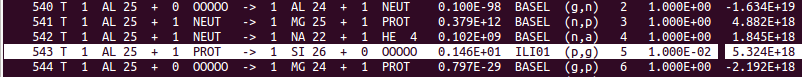
1. import ppn.py module and create an instance of the abundance vector class of data in directory ‘standard’:



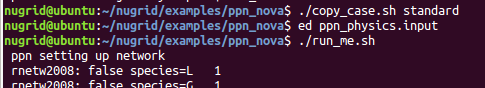
1. In order to perform a plot of an isotopic abundance distribution use the iso\_abund method: 
2. Often we want to do a ratio plot of the abundance at some cycle to some other cycle, especially maybe the zeroth cycle, which contains the initial abundance. This can be accomplished with the ref=nnn option, where nnn is cycle by which we want to divide: 

Next, we would like to perform a reaction rate sensitivity test and plot the result. We would like to know the effect of a reduction of the 25Al(p,γ) by a factor of 100. This can be accomplished by editing the networksetup.txt file. This file is automatically generated during the network setup stage for a new run (ININET=0 in ppn\_physics.input). It lists all species and reactions that are actually going to be used according to the selection criteria specified in ppn\_physics.input. Instead of creating a new networksetup.txt the code can use a previously created networksetup.txt (ININET=3), which may have been modified. Each reaction rate has a column at the end (last but one) that contains a factor (default 1.0) that will be applied to the rate at run time. The second column contains a boolean (T/F) that allows to toggle the rate. There is also a string that indicates the source of the rate, for this case ILI01 (Illiadis 2001), which may be changed in certain ways.

In order to perform the *effect of a reduction of the 25Al(p,γ) by a factor of 100* test we need to change the factor in the networksetup.txt file for that reaction to 0.01:



Since the output of the new run will be written into the present directory we have to use the copy\_case script first to create a new directory for the present run output and copy all relevant data into that directory.



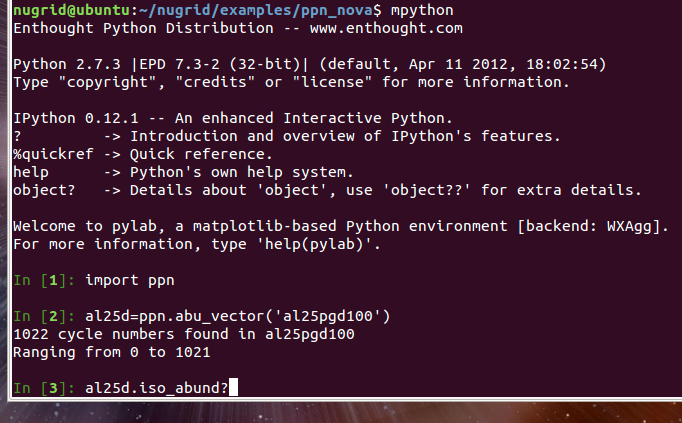
The one-zone code has three input files, one each for the physics, the solver and the frame:



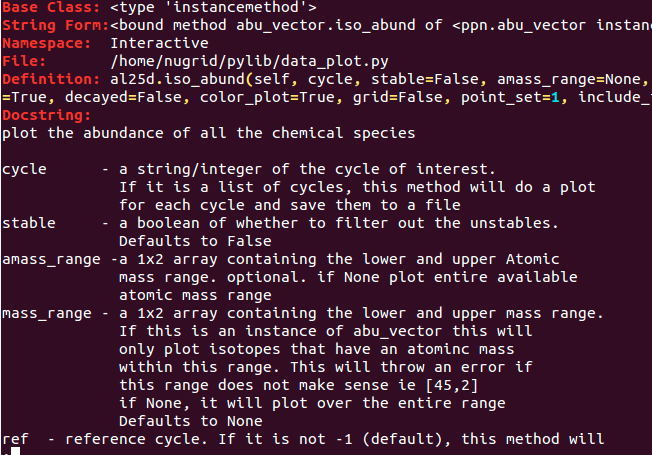
We change ININET to 3 in ppn\_physics.input and execute the run\_script again.

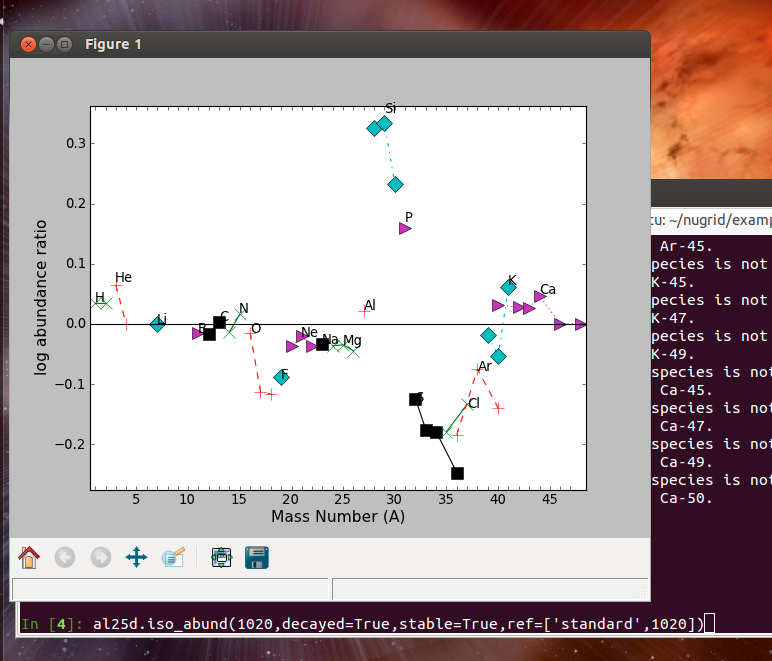
We use the copy\_case.sh script one more time to copy the output to the directory al25pgd100.

In order to visualize the change in the abundance distribution that this reaction rate change has we would like to create a ratio plot of the abundance at the last cycle of the new run divided by the abundance of the last cycle of the standard run. First create a ppn.abu\_vector instance of the run with the changed reaction rate:



Then use again the ref keyword, this time in list mode (we may have forgotten how that works so we first check the doc string):





Now, to get Fig. 15 you need to repeat the same with a run in which you multiply the 25Al(p,γ) rate with 100.